# ECEC 412/621: Computer Architecture
# Solution Set for Quiz 2

Prof. Naga Kandasamy
ECE Department
Drexel University

May 10, 2016

1. Assume the five stage MIPS pipeline with normal forwarding and bypassing capabilities in which branches can be resolved in the ID stage. Also assume that all memory accesses are satisfied in one cycle. Consider the execution of the following loop on this pipeline.

```
Loop:    LD    R2, 0(R1)  ; R2 is the array element
         ADDDI R2, R2, #1 ; Increment R2
         SD    R2, 0(R1)  ; Store result
         DAADI R1, R1, #4 ; Decrement pointer
         BNEZ R2, Loop    ; Branch if last element != 0
```

Schedule the instructions in the loop for best performance including the branch delay slot. Show the timing of the instruction sequence on the pipeline for one iteration of the loop.

The scheduled instructions are as follows:

```
Loop:    LD    R2, 0(R1)   ; R2 is the array element
         DAADI R1, R1, #4  ; Decrement pointer
         ADDDI R2, R2, #1  ; Increment R2
         BNEZ  R2, Loop    ; Branch if last element != 0
         SD    R2, -4(R1)  ; Store result
```

The timing digram for the pipeline:

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| `LD R2,0(R1)` | IF | ID | EX | MEM | WB | | | | | |
| `DAADI R1,R1,#4` | | IF | ID | EX | MEM | WB | | | | |
| `ADDDI R2,R2,#1` | | | IF | ID | EX | MEM | WB | | | |
| `BNEZ R2,Loop` | | | | IF | Stall | ID | EX | MEM | WB | |
| `SD R2,-4(R1)` | | | | | | IF | ID | EX | MEM | WB |

2. Consider the sequence of instructions:

```
LD F6, 34(R2)
LD F2, 45(R3)
MULTD F0, F2, F4
SUBD F8, F6, F2
DIVD F10, F0, F6
ADDD F6, F8, F2
```

The out-of-order scoreboard-based processor has five functional units: two FP multiplication units, one FP divide, one FP add, one integer unit. Assume that each scoreboard stage other than Execute takes one clock cycle. Assume that the `MULTD` and `DIVD` instructions require two and three clock cycles, respectively, to execute and that the `DSUBD` and `ADDD` instructions each take one cycle to execute. The `LD` instruction uses the integer unit to calculate the load address. Assume no forwarding or bypassing hardware but assume that a register read and write in the same clock cycle "forwards" through the register file.

Use the scoreboarding technique and at each clock cycle, trace the handling of each instruction stage-by-stage through Issue (IF), Read Operands (RO), Execute (EX), and Write Result (WB). Provide the corresponding timing diagram.

When generating the timing diagram, note the following key points:

1. Structural and WAW hazards are resolved during the instruction issue stage. If a hazard is detected, then the corresponding instruction is not issued until the hazard is resolved. Also, if an instruction cannot be issued, then subsequent instructions cannot be issued as well, since this is an in-order issue machine. In the code snippet above, if the first `LD` issues at clock cycle 1, the second `LD` cannot be issued at clock cycle 2 since we only have a single integer unit available. Also, none of the subsequent instructions following the second `LD` can be issued.

2. RAW hazards are resolved during the Read Operands stage.

3. WAR hazards are resolved during the Write Result stage. So, though the last `ADDD` finishes before the `DIVD`, we will prevent `ADDD` from overwriting the contents of register F6 until `DIVD` has read its operands.

The following table shows the progress of each instruction through the pipeline during each clock cycle (CC).

| | Issue | Read Operands | Execute | Write Results |
|---|---|---|---|---|
| `LD F6,34(R2)` | CC 1 | CC 2 | CC 3 | CC 4 |
| `LD F2,45(R3)` | CC 5 | CC 6 | CC 7 | CC 8 |
| `MULTD F0,F2,F4` | CC 6 | CC 8 | CCs 9 and 10 | CC 11 |
| `SUBD F8,F6,F2` | CC 7 | CC 8 | CC 9 | CC 10 |
| `DIVD F10,F0,F6` | CC 8 | CC 11 | CCs 12, 13, and 14 | CC 15 |
| `ADDD F6,F8,F2` | CC 11 | CC 12 | CC 13 | CC 14 |

3. Consider the following repeating pattern (e.g., in a loop) of branch outcomes:

T, T, T, NT, NT, ...

where T stands for a taken branch and NT for a not taken branch. What is the accuracy of a two-bit predictor if this pattern is repeated forever? Assume that the predictor is initialized to NT.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Branch outcome | T | T | T | NT | NT | T | T | T | NT | NT |
| Branch prediction | NT | T | T | T | T | NT | T | T | T | T |
| Correct | No | Yes | Yes | No | No | No | Yes | Yes | No | No |

The prediction accuracy is 40%.